FUZZY TOKENS: THINKING CAREFULLY ABOUT TECHNICAL CLASSIFICATION VERSUS LEGAL CLASSIFICATION OF CRYPTOASSETS

David J. Kappos[†]*, Lee A. Schneider*[††]*, Daniel M. Barabander*[†††]*, & Callum A.F. Sproule*[‡]

TABLE OF CONTENTS

I. INTRODUCTION

Popular discourse, when not lumping all cryptoassets together as a homogeneous asset class, has trended towards categorizing blockchain tokens into broad and ill-defined categories. We see this with discussion regarding whether one token or another is fungible versus non-fungible,[1] a security

DOI: https://doi.org/10.15779/Z382J6854B

† Partner, Cravath, Swaine & Moore LLP
†† General Counsel, Ava Labs.
††† Associate, Cravath, Swaine & Moore LLP.
‡ Associate, Cravath, Swaine & Moore LLP.
1. *Fungible vs nonfungible tokens: What is the difference?*, COINTELEGRAPH, https://cointelegraph.com/nonfungible-tokens-for-beginners/fungible-vs-nonfungible-tokens-what-is-the-difference.

versus e-currency,[2] or where tokens are grouped with vastly different assets under terms such as "utility tokens".[3] These shorthand methods of classification oversimplify and distort what tokenization actually accomplishes—namely, enabling the digital representation of anything. These methods also gloss over the diversity of blockchain token functions and how these tokens are used.

For example, the emphasis on fungibility—whether or not a token is unique and irreplaceable—is an accident of technological and regulatory development in the cryptoasset space. It is merely a label used to approximately describe technical coding standards, or a perceived safe harbor where innovation can, at least for the present, outpace regulatory intervention. For instance, while "NFTs" stand for "non-fungible tokens", the NFT label is applied to wildly different assets that possess an array of characteristics. Thus, over-emphasizing non-fungibility at the cost of appreciating other characteristics of a token is short-sighted, counterproductive and breaks down quickly under many common circumstances. A new framework for classification of tokens is necessary.

Rather than categorizing tokens based on their surface-level appearance, our proposed method of "sensible" categorization classifies tokens based on the underlying assets or bundle of rights they represent. Such categorization, in turn, makes regulation more predictable and simpler to implement under existing legal frameworks. It also shows where something new is created that might be susceptible to a more nuanced regulatory approach. To show this, we first examine smart contract technical standards concerning fungibility. We dive into smart contract code to show what token fungibility actually means. We argue that while fungibility is a useful technical concept, it is not an adequate lodestar for classifying tokens for the purpose of applying legal mechanics. We then move beyond technical dichotomies and introduce a more nuanced classification framework that can capture technical and economic realities and facilitate regulation that comports with existing legal frameworks.

2. Wayne Duggan & Farran Powell, *Crypto Regulation: Is Cryptocurrency A Security?*, FORBES (Oct. 7, 2022, 12:02 PM), https://www.forbes.com/advisor/investing/sec-crypto-regulation.

3. Murtuza Merchant, *Utility tokens vs. equity tokens: Key differences explained*, COINTELEGRAPH (May 16, 2022), https://cointelegraph.com/explained/utility-tokens-vs-equity-tokens-key-differences-explained.

II. CASE STUDIES AND ANALYSIS

A. SHIBA INU TOKEN ("SHIB")

SHIB is a popular digital currency that launched in the summer of 2020 as a satirical "memecoin" in response to the success of its namesake, Dogecoin. Unlike Dogecoin, which is a cryptocurrency that exists on its own blockchain,[4] SHIB exists entirely through the state of a smart contract on the Ethereum blockchain.[5] "Smart contract" is a term used to describe autonomously functioning code on a permissionless blockchain—that is, a program or application that relies on the blockchain to execute its commands. While reputedly launched as a joke, the SHIB market cap briefly exceeded $30 billion in late 2021, and continues to hover around $10 billion at the time of this writing.[6]

The SHIB smart contract code follows a standard known as ERC-20, which, among other factors, requires that tokens under the smart contract are finite and transferable.[7] The ERC-20 standard is one of the most commonly used for smart contract tokens, along with ERC-721 and ERC-1155. ERC-20 is also arguably the most basic of these standards, and will serve as a starting point for our analysis. ERC-20 tokens are considered fungible from a technical perspective.

1. Analysis of the SHIB smart contract and ERC-20.

ERC-20 standard smart contracts, such as the contract governing SHIB, contain a code variable that stores information concerning the state of the smart contract, where "state" means the most current information known by the application. In the case of the SHIB smart contract, this variable is named _balances. The _balances variable serves as a ledger that tracks all of the owners of the tokens governed by the smart contract and the quantity of tokens that any given wallet address[8] holds. As such, the state of the SHIB

4. *Dogecoin vs. Bitcoin: Key Differences*, COINTELEGRAPH, https://cointelegraph.com/dogecoin-for-beginners/dogecoin-vs-bitcoin-key-differences (last visited July 21, 2020).

5. Zoltan Vardai, *Shiba Inu Explained — Diving Beyond the Memecoin*, FORKAST (May 2, 2022, 2:24 PM), https://forkast.news/shiba-inu-explained-memecoin.

6. Raynor de Best, *Market Capitalization of Shiba Inu (SHIB) From August 2020 to July 10, 2022*, STATISTA (July 11, 2022), https://www.statista.com/statistics/1271617/shiba-inu-daily-market-cap/.

7. *Id.*; Sina Pilehchiha (@spilehchiha) et al., *ERC-20 Token Standard*, ETHEREUM.ORG (May 22, 2022), https://ethereum.org/en/developers/docs/standards/tokens/erc-20/.

8. Throughout this article, we will refer to addresses as users' wallet addresses; however, this is an oversimplification, as a smart contract address can also be considered the holder of tokens.

smart contract simply lists all accounts and their balances, giving the baseline for a new state once a transfer occurs. The _balances variable is expressed in code as follows:

```
mapping (address => uint256) private _balances;
```

Figure 1: The _balances variable on the SHIB smart contract, which serves as a ledger for the quantity of tokens each address holds.

To demonstrate how SHIB's _balances variable works in practice, we will revisit the remarkable true story of SHIB's launch and earliest transactions, in which SHIB insiders gifted Ethereum co-founder, Vitalik Buterin, over 500 trillion tokens—more than half of SHIB's entire quadrillion token supply.

On August 1, 2020, shortly after the launch of the token's smart contract, wallet address 0x4B4[9] (presumably controlled by a SHIB insider) held 495 trillion SHIB tokens. A second wallet address 0xAb5, widely known to be owned by Buterin, held 10 trillion SHIB (having already been assigned this SHIB immediately after the token's launch).[10] Expressed in code, these balances were recorded in the SHIB smart contract as follows:

```
_balances = {
  "0xAb5": 495000000000000,
  "0x4B4": 10000000000000,
  [Remaining Addresses]: 495000000000000
}
```

Figure 2: The _balances variable on the SHIB smart contract on the morning of August 1, 2020, represented in code.

While the _balances ledger exists only as smart contract code expressed in simple text (as shown above in Figure 2), for illustrative purposes we can represent this ledger through a "lookup table". This lookup table translates the

9. Throughout this paper, all addresses will be abbreviated for the sake of readability.
10. *Shiba Inu Token*, ETHERSCAN, https://etherscan.io/token/0x95aD61b0a150d79219dCF64E1E6Cc01f0B64C4cE?a=0xb8f226ddb7bc672e27dffb67e4adabfa8c0dfa08 (last visited July 21, 2022).

code in a manner we can quickly comprehend, reflecting the key information contained within the _balances variable. Thus, the state of the SHIB _balances variable at this point in time on August 1st, 2020 for wallet addresses 0x4B4 and 0xAb5, can be translated as:[11]

Address	Amount
`0x4B4` (SHIB insider)	`495000000000000`
`0xAb5` (Buterin)	`10000000000000`
All other Remaining Addresses	`495000000000000`

SHIB tokens "exist" insofar as they are accounted for in the lookup table. If numbers are added to this table, new SHIB is created. If numbers are removed from this table, SHIB is destroyed. The smart contract defines the upper limit of tokens that may exist, and the _balances variable indicates how many do exist and how they are assigned to the wallet addresses of each and every token holder in current state.

In addition to its use of the _balances variable, the ERC-20 standard prescribes a function that permits transfers of tokens from one address to another (which would result in new state). On the SHIB smart contract, this function is called transfer(), which is expressed in code as follows:

```
function transfer(address recipient, uint256 amount) . . . {
  _transfer(msg.sender, recipient, amount);
  . . .
}
```

Figure 3: Transfer function on the SHIB smart contract.

We can see this function in action by returning to our example later in the day on August 1, 2020, when the 0x4B4 wallet address called SHIB's transfer() function to send Buterin's 0xAb5 wallet the entirety of its 495 trillion SHIB balance:[12]

11. Wallet Address 0x4B4b7fFb93D8AE84f7e38151F8C6aCBb26605011, ETHERSCAN, https://etherscan.io/address/0x4b4b7ffb93d8ae84f7e38151f8c6acbb26605011 (last visited July 21, 2022).

12. The transfer technically took place over two separate function calls, but we are aggregating them for simplicity.

```
_transfer(0x4B4, 0xAb5, 495000000000000);
```

Figure 4: Transfer function on the SHIB smart contract populated with the information used to transfer 495 trillion SHIB to Buterin's 0xAb5 wallet.

As illustrated, the insider's call to (and thereby use of) this transfer() function set Buterin's wallet address as the recipient address and the amount transferred as 495 trillion SHIB. The transfer() function call updated the information contained within the _balances variable, resulting in new state. This change is reflected in the lookup table as follows:

Address	Amount
`0x4B4` (SHIB insider)	~~`495000000000000`~~ `0`
`0xAb5` (Buterin)	~~`10000000000000`~~ `505000000000000`

Through a simple call to a smart contract function, the SHIB insider transferred approximately $6.7 billion in SHIB (at the time of transfer) to Buterin, who did not need to take a single action to receive the payment.[13] The transfer was accomplished entirely through the updated _balances ledger; nothing changed hands between the transacting parties.

Our retelling of this transaction history illustrates the core purpose of a simplistic but quintessential ERC-20 fungible token. Any one SHIB token is completely interchangeable with the next—nothing distinguishes Buterin's SHIB from the insider's SHIB. This is because the only information tracked by the _balances variable is the quantity of tokens held by any given wallet address. Put differently, SHIB tokens (like all ERC-20 tokens) exist solely as numbers next to addresses on a ledger. They are not digitally unique items. When a transfer of SHIB occurs, the ledger (and therefore the lookup table) simply updates by crediting the line item amount of the transferee and debiting

13. Buterin subsequently burned 410 trillion of his SHIB, and donated 50 trillion SHIB to the India COVID Relief Fund. *See* Vardai, *supra* note 5; Tim Hakki, *Remember All That SHIB Vitalik Buterin Burned? It's Now Worth $32.5 Billion*, DECRYPT (Oct. 28, 2021), https://decrypt.co/84645/remember-all-that-shib-vitalik-buterin-burned-now-worth-32-5-billion.

that of the transferor. The traditional world analogue is the way dollars exist on bank computers—in people's accounts. This differs from paper money, which is represented by individual physical items that are unique (but fungible).

By deconstructing the functionality of a basic ERC-20 token, we have built a foundation to understand not only the ERC-721 and ERC-1155 standards, but also examine novel uses of blockchain tokens that challenge our ability to categorize them as fungible or not.

B. DOODLES

Doodles is a popular profile picture ("PFP") NFT project that launched in October 2021,[14] with a market cap that quickly climbed into the hundreds of millions of dollars.[15] Like many popular PFP projects, Doodles NFTs depict a series of themed art assets, with variable traits that contain different levels of rarity assigned to individual tokens. In contrast to the quadrillion tokens minted for SHIB, the Doodles collection is comprised of just 10,000 tokens and corresponding art assets. Unlike the ERC-20 SHIB smart contract, the Doodles smart contract implements a standard, known as ERC-721.

Figure 5: Eighteen examples of the 10,000 art assets that comprise the Doodles PFP NFT series.

14. Reethu Ravi, *Doodles NFT: The Only Guide You Need*, NFT EVENING (Feb. 13, 2022), https://nftevening.com/how-doodles-became-one-of-the-nft-industrys-most-loved-collections.

15. Doodles, WORLDCOININDEX, https://www.worldcoinindex.com/nft/doodles (last visited July 21, 2022).

The ERC-721 standard is used to create non-fungible tokens.[16] As with ERC-20, tokens minted under the ERC-721 standard are finite and transferable. However, whereas the ERC-20 ledger tracked the quantity of tokens assigned to each wallet address, the ERC-721 ledger tracks individual tokens, with wallet addresses assigned to unique token index numbers. Each index number corresponds to one token—for instance, the Doodles ledger contains index numbers ranging from 1–10,000, one for each of the tokens in the series. Unlike ERC-20, wallet addresses in ERC-721 are not assigned a quantity of tokens (as each index number corresponds to only one token), but rather are assigned to one or more index numbers to indicate ownership of any given token.

Section [II.B.1] demonstrates how ERC-721 implementations keep track of token ownership by examining the code of the Doodles smart contract and constructing a lookup table to illustrate the key characteristics of the ERC-721 standard. Section [II.B.1.] then introduces complexities that frustrate simplistic categorization.

16. Hursit Tarcan (@hursittarcan) et al., *ERC-721 Non-Fungible Token Standard*, ETHEREUM.ORG (June 23, 2022), https://ethereum.org/en/developers/docs/standards/tokens/erc-721/.

1. Analysis of the Doodles smart contract and ERC-721.

When Doodles were initially offered for sale, users purchased them through the following mint function:

```
contract Doodles is ERC721, ERC721Enumerable, Ownable {

  . . .

  function mint(uint numberOfTokens) public payable {
    uint256 ts = totalSupply();
    . . .
    for (uint256 i = 0; i < numberOfTokens; i++) {
      _safeMint(msg.sender, ts + i);
    }
  }

  . . .

}
```

Figure 6: The mint() function on the Doodles smart contract, used to create the initial supply of Doodles.

For our purposes, the key line contains the _safeMint() function. Just as we saw with SHIB, the Doodles smart contract stores who owns what tokens through a ledger, which we will again reimagine as a lookup table for ease of reference.[17] The _safeMint() function adds new tokens to this lookup table as the act of initial creation.

To understand _safeMint()'s operation, imagine a hypothetical user with the wallet address 0xABC. This user wants to purchase one Doodle, which so happens to be the 9,999th Doodle of the series of 10,000. The user will call mint() and pass in the numberOfTokens she would like to mint as 1.

17. Note that whereas SHIB called its ledger variable _balances, Doodles calls the ledger variable _owners—the variation in the variable's name is idiosyncratic from contract-to-contract and not material to its function.

```
_safeMint(0xABC, 9998 + 1);
```

Figure 7: The populated _safeMint() function that is executed to mint() Doodle 9,999 for wallet address 0xABC.

Consequently, the lookup table updates to reflect her ownership:

Index of Token	Address
9999	0xABC

Next, assume user 0xXYZ also wants a Doodle. Again, the user calls mint(), which this time populates as follows:

```
_safeMint(0xXYZ, 9999 + 1);
```

Figure 8: The populated _safeMint() function that is executed to mint() Doodle 10,000 for wallet address 0xXYZ.

The lookup table updates accordingly:

Index of Token	Address
. . .	. . .
9999	0xABC
10000	0xXYZ

Next, assume that user 0xXYZ wishes to transfer her NFT with index number 10,000 to 0xABC. This is achieved by user 0xXYZ calling a separate transferFrom() function:

```
function transferFrom(
  address from,
  address to,
  uint256 tokenId
) public virtual override {
  . . .
  _transfer(from, to, tokenId);
}
```

Figure 9: The transferFrom() function, which Doodle holders use to transfer their NFTs from one holder to another.

This passes in the following values:

```
_transfer(0xXYZ, 0xABC, 10000);
```

Figure 10: The populated transferFrom() function, which holders use to transfer their Doodles from one address to another.

Note how similar the Doodles transferFrom() function looks to the SHIB transfer() function used in the 495 trillion SHIB Buterin transaction. The material difference between the two is that while the SHIB function call was populated with the quantity of SHIB to be transferred (495 trillion SHIB), the Doodles function call is populated with the index number of the individual token to be transferred (10,000). In both instances, the transfer is executed by a simple update of the smart contract ledger that updates state. Accordingly, the Doodles lookup table will now look as follows:

Index of Token	Address
. . .	. . .
9999	0xABC
10000	~~0xXYZ~~ 0xABC

Doodles are non-fungible because each Doodle token is represented by a unique index number, assigned to one specific address. Put simply, a token with an index number of 9999 is not entirely interchangeable on the ledger with a token with index number of 10000. It is easy to overstate the difference between ERC-20 fungible and ERC-721 non-fungible standards. At their core, the former tracks token quantity and the latter tracks individual token identity. Put another way, current state shows balances for ERC-20 and digitally unique items for ERC-721. All other characteristics assigned to one standard or the other in popular discourse are, from a technical perspective, superfluous.

This revelation has implications for how we might think about fungibility generally. On the one hand, if the Doodle art asset tied to index 9999 was identical to that tied to index 10000 (that is, if they referenced the exact same artwork), then from a strict technical perspective, these tokens would still be non-fungible because of their unique index identifiers. Nevertheless, from a practical standpoint, their valuation or utility to any given purchaser may be indistinguishable. If a creator were to create an entire series of tokens referencing 1,000,000 identical photographs (or even no item at all), then we can imagine that series of tokens being as fungible as physical dollar bills, despite having been minted through the ERC-721 non-fungible standard. Conversely, consider a series of ERC-20 tokens that grant specific permissions to users who hold a certain percentage of outstanding tokens. While in a technical sense the tokens remain non-fungible, in a practical sense they are fungible tokens, assigned new powers based upon whether they are held in sufficient quantity by a controlling account. Bifurcation along lines of fungibility becomes even more muddied when we turn to the ERC-1155 standard, analyzed next.

C. ADIDAS ORIGINALS INTO THE METAVERSE AND ERC-1155

As discussed above, ERC-20 tokens such as SHIB are characterized as fungible, while ERC-721 tokens such as Doodles are non-fungible. In contrast, ERC-1155 standard tokens incorporate aspects of both ERC-20 and ERC-721 tokens and are sometimes described as semi-fungible. Examining a smart contract that implements ERC-1155, such as the smart contract utilized by the Adidas Originals into the Metaverse project (the "Adidas Project"), presents an illustration as to why fungibility is a sliding scale that cannot be reliably used for token categorization.

1. Background on the Adidas Project

The Adidas Project is a collaboration between Adidas Originals and NFT projects gmoney, Bored Ape Yacht Club and PUNKS Comic. The Adidas Project token grants holders access to physical merchandise and promises

access to certain metaverse content.[18] The Adidas Project originally sold 30,000 tokens constituting the collection. From April 28, 2022 to May 18, 2022, token holders could redeem their tokens in exchange for physical Adidas Project merchandise, such as hoodies and beanies.

The Adidas Project dubs its token the "Into the Metaverse (ITM) NFT" and consistently uses the term "NFT" throughout its marketing.[19] The Adidas Project website also notes that the Ethereum blockchain ensures "one NFT only has one owner at any given time."[20] For instance, ahead of its launch on OpenSea, Adidas tweeted: "All . . . Into the Metaverse NFTs have now been minted."[21] The Adidas Project OpenSea listing describes the collection as a "collaborative, co-created NFT project between Adidas Originals and NFT pioneers . . . "[22] Following suit, news articles reporting and publicizing the Adidas Project have also adopted the term "NFT" in describing the Adidas Project tokens.[23]

Given such marketing, we would expect the Adidas Project tokens to have non-fungible characteristics, similar to those we outlined above for Doodles. Namely, we might expect that only one wallet address would be assigned to any given unique index number on a ledger contained within the smart contract, which would be consistent with the characteristics of many other tokens with the NFT label. Despite this expectation, examining the Adidas Project smart contract code demonstrates a different result.

18. ADIDAS INTO THE METAVERSE, https://www.adidas.com/into_the_metaverse (last visited July 21, 2022).

19. *Id.*

20. *FAQ*, ADIDAS INTO THE METAVERSE, https://www.adidas.com/metaverse/faq (last visited Jan. 10, 2023).

21. Adidas Originals (@adidasoriginals), TWITTER (Dec. 17, 2021, 6:26 PM), https://twitter.com/adidasoriginals/status/1471985083280199680.

22. *Adidas Originals Into the Metaverse*, OPENSEA, https://opensea.io/collection/adidasoriginals (last visited July 21, 2022).

23. Jacob Kastrenakes, *Adidas is Launching an NFT Collection With Exclusive Access to Streetwear Drops*, THE VERGE (Dec. 16, 2021), https://www.theverge.com/2021/12/16/22822143/adidas-nft-launch-into-the-metaverse-price-release-date; Rima Sabina Aouf, *Adidas to Enter the Metaverse with First NFT Products*, DEZEEN (Dec. 19, 2021), https://www.dezeen.com/2021/12/19/adidas-enter-metaverse-first-nft-products-design/.

2. Technical analysis of the Adidas Project smart contract.

a) Purchasing the token.

To begin, we examine the _purchase() function that governs the initial token purchase on the Adidas Project smart contract:[24]

```
contract AdidasOriginals is AbstractERC1155Factory,
PaymentSplitter {

  . . .

  function _purchase(uint256 amount) private {
    . . .
    _mint(msg.sender, 0, amount, "");
    . . .
  }

  . . .

}
```

Figure 11: Adidas Project smart contract _purchase() function, used for the initial mint.

When a user purchases a token, she initiates a call to the _purchase() function, as illustrated in Figure 11. The _purchase() function updates the smart contract's ledger, named _balances, with the quantity of tokens the user has purchased. It does this by calling _mint(). To illustrate how _mint() works on this smart contract (and how it differs from Doodles' (_safeMint()), consider a hypothetical transaction in which a user with the wallet address 0xABC initiates a call of the _purchase() function requesting one token.[25] This transaction is reflected on a constructed _balances lookup table as follows:

24. AdidasOriginals Smart Contract, ETHERSCAN, https://etherscan.io/address/0x28472a58a490c5e09a238847f66a68a47cc76f0f#code (last visited July 21, 2022).

25. In this example, the user's purchase calls the _mint() function using the following values in brackets: _mint(msg.sender [0xABC], 0, amount [1], "").

Index of Token	Address	Amount
0	0xABC	1
	. . .	. . .

Next, in a separate transaction, assume a second user with the wallet address 0xXYZ calls the _purchase() function to purchase two additional tokens.[26] Consequently, the _balances ledger updates, and the lookup table adjusts:

Index of Token	Address	Amount
0	0xABC	1
	0xXYZ	2

Of note, the tokens purchased by user 0xABC and 0xXYZ are both referenced by the same index number (*i.e.*, the index number 0). Thus, the only distinguishing variable assigned to either wallet address is the quantity of tokens purchased. For this reason, and as we saw with SHIB in the ERC-20 context, the tokens generated for these two users are indistinguishable. Put differently, they are fungible.

b) Phases and redeeming tokens.

The question arises: Why use ERC-1155 tokens at all? In the case of the Adidas Project, the token index is universally assigned to 0 at the time of the token's initial purchase and is subsequently used to track the "phase" of any given token. A token's phase corresponds to whether or not it has been redeemed for physical merchandise. Tokens that have not been redeemed for merchandise retain an index of 0, as assigned at purchase. Tokens that have been redeemed are assigned a new index number that signifies that they have already been used to claim merchandise. To illustrate, we return to our earlier hypothetical purchase, in which the _balances lookup table currently stores the following information as state:

26. The _mint() function look as follows: _mint(msg.sender [0xXYZ], 0, amount [2], "").

Index of Token	Address	Amount
0	0xABC	1
	0xXYZ	2
	. . .	. . .

When user 0xABC chooses to redeem her token for merchandise during a redemption period, she calls the redeemCardForOther() function illustrated below:

```
function redeemCardForOther(uint256 cardIdToRedeem, uint256
amount) . . . {
  . . .
  _burn(msg.sender, cardIdToRedeem, amount);
  _mint(msg.sender, cardIdToMint, amount, "");
  . . .
}
```

Figure 12: Core logic for "redeeming" tokens on the Adidas Project smart contract.

Upon calling the redeemCardForOther() function, a series of checks will first ensure that the user has a token with an index of 0 available to redeem; we have excluded these checks from Figure 12 to simplify the analysis.[27] We can see in the _balances lookup table that user 0xABC has a token with an index number of 0, so she passes these initial checks. Once the checks are successfully satisfied, the function next calls the _burn() line of code. _burn() is the inverse of _mint(), in that it reduces rather than increases the number of

27. This function first checks that the user calling the function has the quantity at the token index that she is passing in for cardIdToRedeem on the _balances lookup table (require(balanceOf(msg.sender, cardIdToRedeem) >= amount). Because our user has 1 on the lookup table for token index of 0 and is only trying to claim 1, the check will pass.
The function checks that the token index the user is redeeming is less than the phase of the project (require(cardIdToRedeem < cardIdToMint)). At the time of this writing, the project phase (cardIdToRedeem) is 1, and our user's token index (represented by cardIdToRedeem, but we know the user has it because the previous step was passed) is 0, so this check will also pass.

tokens assigned to a wallet address. Consequently, the _balances lookup table updates state as follows:[28]

Index of Token (Phase)	Address	Amount
0	0xABC	~~1~~ 0
	0xXYZ	2
	. . .	. . .

The index 0 token has been removed from user 0xABC's wallet address. In its place, a new token is created and assigned to 0xABC's wallet by calling the _mint() function.[29] Unlike the call to the minting function used for the initial purchase, the redemption _mint() function call uses the current marketing phase as defined by the smart contract.

The net effect of burning and minting in this instance is that an index 0 token is replaced with an index 1 token in new state:

Index of Token (Phase)	Address	Amount
0	0xABC	0
	0xXYZ	2
	. . .	. . .
1	0xABC	1
	. . .	. . .

After redeeming the token, merchandise is sent to user 0xABC through off-chain logistics. Like the index 0 token, the new index 1 token is totally fungible with all other tokens that share its index number. Put differently, there are now two distinct series of fungible tokens governed under a single smart contract. This explains why, despite there being 30,000 of the Adidas Project "NFTs" in circulation, only two types (respectively corresponding to index 0 and index 1) can currently be distinguished on OpenSea.[30]

---

28. In this example, the user calls the _burn() function using the following values in brackets:
_burn(msg.sender [0xABC], cardIdToRedeem [0], amount [1]).

29. In this example, the user calls the _mint() function using the following values in brackets:
_mint(msg.sender [0xABC], cardIdToMint [1], amount [1], "").

30. Adidas Originals: Into the Metaverse, OPENSEA, https://opensea.io/asse ts/ethe reum/0x28472a58a490c5e09a238847f66a68a47cc76f0f/0 (last visited July 21, 2022).

The question remains: Where does non-fungibility come into play that makes these tokens NFTs? While the Adidas Project tokens are non-fungible on the index level (which defines the membership marketing "phase"), nothing is unique about each given holder's membership within that index. In our hypothetical, the only way a non-fungible token could theoretically be minted would be if user 0xABC happened to be the only user who chose to redeem her index 0 token and mint an index 1 token, or, alternatively, that all other users had redeemed their index 0 tokens for index 1 tokens such that hers was the only remaining index 0 token.[31] In such an instance, this sole index 1 token (or the sole index 0 token) would be unique and non-fungible. However, nothing in the Adidas Project smart contract restricts the number of tokens that may be issued within any one index number. Indeed, at the time of this writing, 5,822 identical index 0 tokens and 24,178 identical index 1 tokens are in circulation.[32] Fungibility, therefore, constitutes a poor characteristic for purposes of categorization. A better classification system is needed.

III. APPLYING A SENSIBLE CLASSIFICATION SYSTEM

As this Commentary outlines, relying on labels of technical standards like ERC-20, ERC-721 or ERC-1155 to categorize blockchain tokens tends to oversimplify and mischaracterize them. This is because technical standards, while useful to understand the coding mechanics of a smart contract, can easily be utilized to create tokens with unanticipated and even counterintuitive purposes that defy preconceived characteristics such as fungibility. A sensible classification system recognizes that smart contracts and tokens are tools with an unlimited number of potential uses, as illustrated above. The sensible taxonomy resists surface-level characterizations and instead scrutinizes actual token functionality and feature set to determine the reality of the bundle of rights represented by the token. When we acknowledge that a token is simply a digital representation of something, whether tangible or intangible, we understand that the token should be classified in accordance with the functionality and features of the thing it represents. This function-first mindset comports with how the U.S. federal securities laws approach, for example, the dominant "economic reality" framework of investment contract analysis,

31. *See, e.g.*, ERC1155,OPENZEPPELIN, https://docs.openzeppelin.com/contracts/3.x/erc1155 (last visited July 21, 2022) (where a token is labeled as "non-fungible" because only one token was minted for one address at a unique index).

32. Determined by calling totalSupply() for token index number 0 (marketing phase 1) and token index number 1 (marketing phase 2) on the smart contract directly. *See* AdidasOriginals Smart Contract, ETHERSCAN, https://etherscan.io/address/0x28472a58a490c5e09a238847f66a68a47cc76f0f#readContract (last visited July 21, 2022).

under which arrangements concerning the sale of chinchillas, whiskey warehouse receipts, oyster beds and live silver foxes might constitute an investment contract, even while those items themselves are not securities.[33]

In essence, a sensible taxonomy examines whether a token represents: (1) an underlying physical asset; (2) a credit for the performance of a service or limited licensing right; (3) an intangible asset such as a security, intellectual property or representation of real estate holdings; (4) a stablecoin pegged to a fiat currency; or (5) some integral component of a distributed ledger blockchain or smart contract (a "Native DLT Token").[34] In short, the core functions and feature set of the token should, rightly, serve as the basis for the legal and regulatory classification. By classifying tokens in this manner, we can largely use existing and well-developed legal and regulatory regimes to govern them.

To put this sensible taxonomy to use, consider our three exemplary NFT regimes: SHIB, Doodles and the Adidas Project. SHIB, the most simplistic of our three examples (both in terms of its smart contract code and the token's underlying function) is, in its paradigmatic form, a straightforward example of a Native DLT Token.[35] This is because SHIB represents ownership of a fully digital, distributed, ledger-based currency, which exists through the smart contract that governs it and does not reference or entitle the holder to any other asset or item. The token is not an abstraction of an underlying asset, but rather a direct representation of the asset itself and does not exist without the smart contract.

Doodles presents a more complex scenario and serves as an illustrative test case where areas of categorization become blurred. At first glance, Doodles appears to constitute a limited-use license for digital art assets, and thus an example of a services token. This is because the purchaser of a Doodles NFT is granted limited rights to display and monetize the associated Doodles art asset.[36] The purchaser and the art asset's IP owner have a licensee/licensor

33. SEC v. W.J. Howey Co., 328 U.S. 293 (1946); *cf.* Gary Gensler, Chair, Sec. & Exch. Comm'n, Prepared Remarks of Gary Gensler on Crypto Markets Penn Law Capital Markets Association Annual Conference (Apr. 4, 2022), https://www.sec.gov/news/speech/gensler-remarks-crypto-markets-040422.

34. For a fully articulated explanation of this taxonomy, *see* Lee A. Schneider, *A "Sensible" Token Classification System*, CHAMBERS GLOBAL PRACTICE GUIDES, FINTECH 2022 (Mar. 24, 2022), https://practiceguides.chambers.com/practice-guides/fintech-2022.

35. *A Sensible Token Classification System*, NOVUM INSIGHTS (June 9, 2021), https://novuminsights.com/post/sensible-token-classification-system/.

36. *See Terms of Service*, DOODLES, https://docs.doodles.app/terms-of-service (last visited July 21, 2022) ("By connecting your Ethereum wallet and minting an NFT with our smart contract, you have purchased a Doodle! With this Doodle you can show it off, use it as your pfp, sell it, and even merchandise it up to $100,000 through the sale of physical merch

relationship with regard to the art asset. This limited license arrangement is similar to the relationship between users and IP owners in music streaming services, in which users have a limited use license to play an IP owner's music. However, a Doodle is more than a piece of art providing certain consummate IP rights.[37] Owning a Doodle also entitles its holder to vote on proposals (albeit, off-chain) for Doodles community initiatives, with one Doodle providing its holder with one vote.[38] Proposals have ranged from the (financially) immaterial (such as whether to give a community member a hug)[39] to the consequential, such as how to spend the substantial community treasury funds to grow the project.[40] With these voting rights in mind, the token straddles the line between services token and intangible asset token—the latter constituting a right to participate in the project's decentralized autonomous organization ("DAO") governance structure. Analysis and categorization, therefore, must be contingent on the token's actual use. When it is used for its licensing properties, the token must be treated as a services token, and when used to exert voting power in the DAO, as an intangible asset token.

Whereas Doodles presents a scenario in which a token straddles between multiple classifications, the Adidas Project is an example in which the token's function evolves over the life of a project through the replacement of the original token with a new token once the original is burned. The Adidas Project token begins as a physical asset token, then, with the change in index number, comes to represent an intangible asset as the holder exercises the right to receive the physical item(s) and the project matures through new features supplied by the creator. Simplistic bifurcation based on, for instance, fungibility, is incapable of acknowledging the token's evolving use. The

---

or using your full Doodle in a piece of art you may create. Should you approach that number or expect to go beyond it please reach out to the team and we'll discuss a licensing deal for anything beyond that amount.").

37. For example, the image of Doodle #1 is stored at: https://ipfs.io/ipfs/QmTDxnzcvj2p3xBrKcGv1wxoyhAn2yzCQnZZ9LmFjReuH9 (last visited Feb. 6, 2023).

38. Doodles, OPENSEA, https://opensea.io/collection/doodles-official (last visited July 21, 2022) ("Each Doodle allows its owner to vote for experiences and activations paid for by the Doodles Community Treasury."); Doodles Proposals, SNAPSHOT, https://snapshot.org/#/doodles.eth (last visited July 21, 2022).

39. @bools.eth, Give Bool a Hug, SNAPSHOT (Jan. 14, 2022, 4:34 PM), https://snapshot.org/#/doodles.eth/proposal/0xb4bb9200e1743d14ed903fd0a55f523305e652f71f2
cc394d2dd1487fa2f9122.

40. @0x2B3Ab8e7BB14988616359B78709538b10900AB7D, [FINAL] Team Scaling v2, SNAPSHOT (Mar. 8, 2022, 3:39 PM), https://snapshot.org/#/doodles.eth/proposal/0x23c4ad20c26cc3ce0f3989f75e53cc6089e33af44e17144828d7ac6ab6c71f55.

sensible classification system, in contrast, may be used to re-analyze tokens over the course of a project's entire life span.

To apply the sensible classification system to the Adidas Project, we must look at the current state of the token at applicable points in time, as it changes through holder action and in accordance with the core functionality of the smart contract. Upon issuance, the token initially functioned as a physical asset token. At that early stage, its sole function was to provide users with a mechanism to claim limited edition physical merchandise. Subsequently, once the initial merchandise offering phase expired, the Adidas Project marketing materials suggest that the tokens may be used for not only future merchandise drops, but also non-merchandise membership benefits, such as "virtual land experiences" in a metaverse platform and live events.[41] Thus, once the original token is burned through redemption and the new token created in its place, the token's classification changes as its purpose evolves. The Adidas Project shows how tokens are merely the foundation on which to build—what is built, and thus how the token is categorized, is often in the hands of the project's creator rather than based on any discrete characteristic of the token. A sensible categorization system looks beyond non-essential characteristics to the underlying use even as it evolves to capture the token's core purpose.

IV. CONCLUSION

Dichotomies such as fungible and non-fungible are admittedly convenient shorthand to describe the technical classification of a digital asset. However, these simplifications are ultimately insufficient to give full color to the nature of a token in light of the vast array of potential applications and specific cases in currently existing blockchain applications, let alone future innovations.[42] This shorthand obscures the utility underlying a given token, reducing it to nothing more than a lookup table.

In contrast, our proposed system of categorization leverages existing legal classification structures under the premise that a tokenized asset should be treated in the same manner as the bundle of rights it represents. This sensible system of classification thereby more accurately characterizes the technical function of these different assets. The simple truth is that labels matter, and can change how people interact with assets and how agencies regulate them.

41. ADIDAS INTO THE METAVERSE, *supra* note 18.

42. *Cf. Defining Digital Assets, DACS: The Digital Asset Classification Standard*, COINDESK (Jan. 3, 2023), https://www.coindesk.com/markets/2022/12/15/dacs (classifying digital assets by sector, industry group and industry).

Classifying assets based on one technical characteristic at the expense of all others fails to appreciate the various ways in which tokens can function.